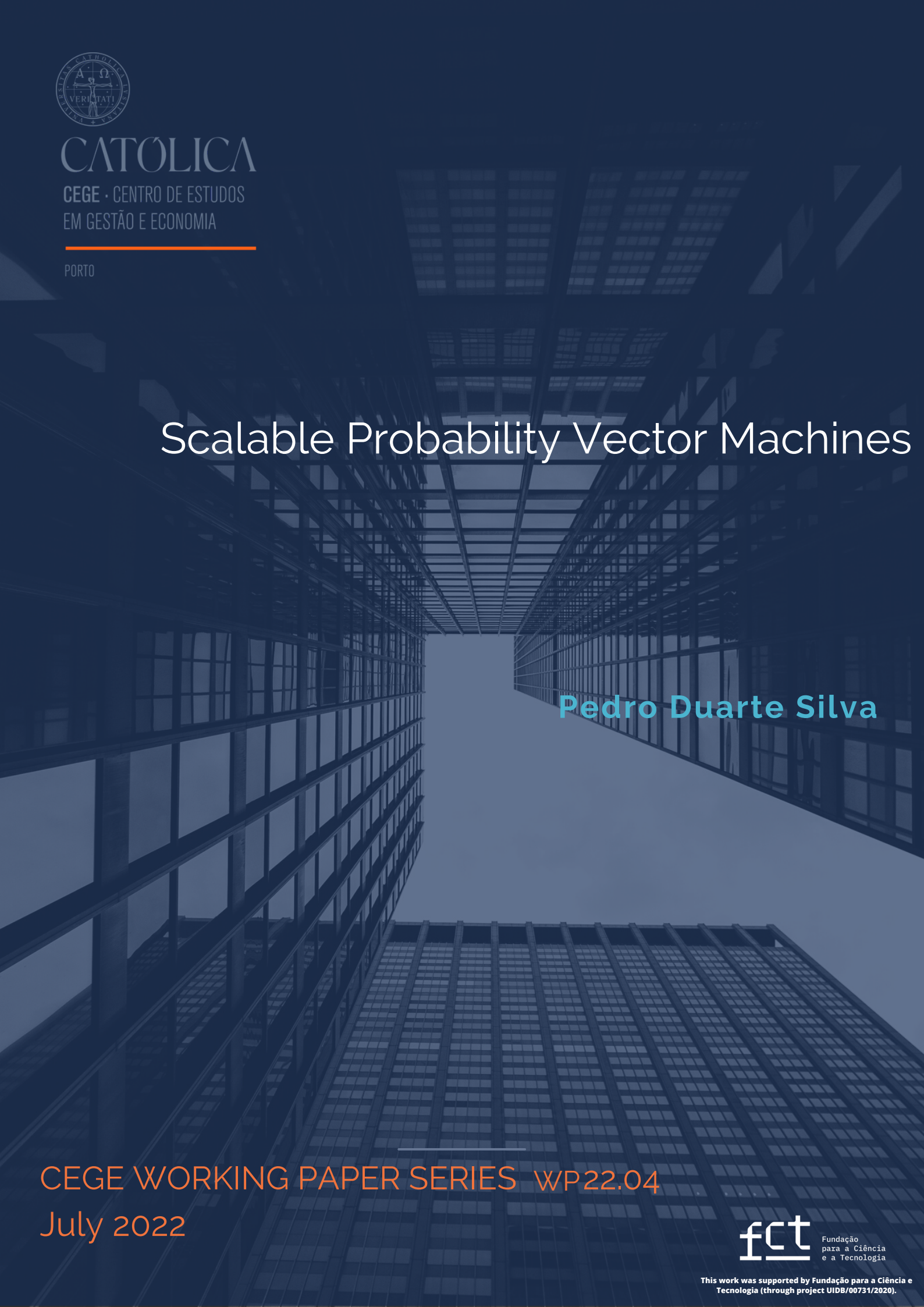

CATÓLICA

CEGE · CENTRO DE ESTUDOS EM GESTÃO E ECONOMIA

PORTO

Scalable Probability Vector Machines

Pedro Duarte Silva

CEGE WORKING PAPER SERIES WP22.04

July 2022

Fundação para a Ciência e a Tecnologia

This work was supported by Fundação para a Ciência e Tecnologia (through project UIDB/00731/2020).

Scalable Probability Vector Machines

A. Pedro Duarte Silva
Católica Porto Business School & CEGE
Universidade Católica Portuguesa

July 29, 2022

Abstract

Over the last few decades, kernel based methods have added an important set of tools to any statistician or econometrician, providing competitive, model free, alternatives to traditional methodologies. In particular, classification Support Vector Machines (SVMs) have proven to be among the most accurate predictors of unknown class memberships in supervised classification problems. Unfortunately, standard SVMs do not complement these predictions with reliable estimates of the corresponding class membership probabilities. Recently, it has been shown that sequences of nonstandard (weighted) SVMs can overcome this limitation, and may be used to recover precise class probability estimates. However, existing implementations of this approach can be computationally too demanding, and may not scale well when the number of different classes grows. In this work, we will present an improved method to build k-class probability estimates from sequences of weighted SVMs, with good scaling properties as k increases. Simulation experiments show that class probability estimation based on weighted SVMs is often more accurate than competing distribution free machine learning approaches, and are more reliable than multinomial logisic regression when its assumptions fail. A public domain R package implementing this proposal is under preparation.

Key Words and Phrases: support vector machines, kernel methods, multiclass classification, multiclass probability estimation.

1 Introduction

Support Vector Machines (SVMs) were originally designed to handle two-class classification problems, and have quickly established themselves as one of the most accurate machine learning algorithms for class prediction. However, this success did not translate to the related task of deriving confidence measures for these predictions, such as reliable probability estimates of class membership. In fact, Lin (2002) has shown that by targeting directly classification boundaries, standard SVMs do not carry much further information

about class probabilities other than the predicted class by itself. Nevertheless, Wang, Shen and Liu (2008) noted that by appropriately weighting the loss function used in standard SVMs, nonstandard SVMs can consistently estimate a theoretical Bayes rule for any arbitrary setting of class probabilities. Based on this property, Wang et al. (2008) proposed to solve sequences of weighted SVMs with varying weight specifications, and to recover class probabilities from the frontiers between regions of the weight domain that lead to different predictions. This proposal was generalized by Wu, Zhang and Liu (2010) to the general multi-class classification problem. However, for this purpose, several difficulties needed to be resolved.

Firstly, in general multi-class problems the recovery of class probabilities from the solutions of weighted SVMs is not straightforward, and direct generalizations of the corresponding two-class procedure may lead to unstable estimates. To correct this problem, Wu et al. (2010) proposed a semi-analytical indirect procedure based on the observed class prediction frequencies, and applied it successfully to several 3-class and 5-class problems. However, in this procedure, the number of base weighted SVMs that need to be trained increases exponentially with the number of classes, and we are not aware of any application of this approach to problems with more than 5-classes.

Secondly, consistency properties are not guaranteed for general multi-class classification SVMs, and many of the best known multi-class SVMs, particularly the popular Crammer and Singer (CS) SVM (Crammer and Singer, 2001), do not satisfy it. To correct this problem, Wu et al. (2010) modified the loss function used in the CS SVM in such a way that consistency is satisfied. Unfortunately, this modification had the unintended consequence of leading to SVMs whose training requires the optimisation of nonconvex problems, increasing the computational challenges, and making this method impractical for big, or even moderate, data problems.

This paper focus on strategies designed to overcome these computational difficulties. In particular, on the one hand, we will propose an improved method for recovering class probability estimates from weighted SVM predictions. In our approach, these estimates will be based on the solution of linear programming models that optimize an l_1-norm measure of the agreement between the predictions implied by probability estimates, and those made by weighted SVMs. One important advantage of this strategy is that, unlike in Wu et al. (2010) method, the different weight specifications used do not have to be uniformly distributed over a k-dimensional simplex, which allows for the creation of grids with satisfatory resolution, while ensuring that the number of required weighted SVMs only grows linearly with the number of different classes. On the other hand, we propose to replace Wu, Zhang and Liu (WZL) SVM by an SVM based on a universal kernel without bias terms, using the weighted loss proposed by Lee, Lin and Wahba (LLW) (Lee et al. (2004)). We note that the LLW loss leads to convex optimization problems and, as noted by Dogan, Glasmachers and Igel (2016), for multiclass SVMs based on universal kernels, dropping bias terms is statistically of minor importance, while allowing for the use of computationally efficient decomposition training algorithms.

Based on these strategies, we were able the find reliable class probability estimates for several problems with hundreds of observations and dozens of different classes. Simulation experiments show that the resulting estimators are competitive, comparing favorably with

alternative machine learning based approaches, as well as multinominal logistic regression when its assumptions are violated.

The remainder of this paper is organized as follows. Section 2 introduces notation, and presents the weighted multiclass SVM formulations used in the paper. Section 3 formalizes the method of estimating class estimates from sequences of weighted SVMs. Section 4 presents an efficient training algorithm for these SVMs. Section 5 describes controlled simulation experiments, and Section 6 concludes the paper.

2 Weighted Multiclass SVMs

Let $\mathcal{T} = \{(x_1, y_1), \ldots, (x_n, y_n)\}$ be a training set of n examples, where x_i is an attribute descriptor belonging to some domain, $\mathcal{X}$, the label y_i is an integer belonging to the set $\mathcal{Y} = \{1, \ldots, k\}$, and all pairs (x_i, y_i) were independently generated from some unknown, but common, probability distribution, $P(\mathbf{X}, Y)$. Furthermore, let $\mathcal{E} = \{x_1, \ldots, x_m\}$ be an estimation set of m examples with unknown labels. Based on $\mathcal{T}$, we are interested in developing estimator functions $(\mathbf{p}_c(\mathbf{x}) \; ; \; c \in \mathcal{Y})$ for the *posterior* class probabilities:

$$\mathbf{p}_c(\mathbf{x}) = P(Y = c | \mathbf{X} = \mathbf{x}) = \frac{P(\mathbf{x}, c)}{\sum_{c' \in \mathcal{Y}} P(\mathbf{x}, c')} \tag{1}$$

and apply these functions to all $x_i \in \mathcal{E}$. Such estimators are to be recovered from a sequence of nonstandard (weighted) multi-class SVMs yielding decisions rules with general form

$$\hat{y} = argmax_c \; \mathbf{f_c}(\mathbf{x}) \tag{2}$$

where $\mathbf{f_c}$ is the c^{th} element of the vector function $\mathbf{f} : \mathcal{X} \mapsto \mathbb{R}^k$ that solves the optimization problem:

$$min_{\mathbf{f} \in \mathcal{F}^k} \quad n^{-1} \sum_{i \in \mathcal{T}} \boldsymbol{\pi}_{y_i} L(\mathbf{f}(\mathbf{x}_i), y_i) + \lambda J(\mathbf{f}) \tag{3}$$

$$subject\ to \qquad \sum_{c \in \mathcal{Y}} \mathbf{f}_c = \mathbf{0} \tag{4}$$

Here, $\mathcal{F}^k$ is a cartesian product of some known functional space, $\mathcal{F}$, the penalty operator $J : \mathcal{F}^k \mapsto \mathbb{R}_0^+$ measures model complexity, $\lambda \in \mathbb{R}^+$ is a regularization parameter that controls the trade-off between the smoothness of $\mathbf{f}$ and the multi-class large margin loss $L : \mathbb{R}^k \times \mathcal{Y} \mapsto \mathbb{R}_0^+$, and the weighting vector, $\boldsymbol{\pi}$, belongs to the k-dimensional simplex, $A_k = \{\boldsymbol{\pi} \in \mathbb{R}^k : \sum_{c=1}^k \boldsymbol{\pi}_c = 1 \; , \; \forall c \; \boldsymbol{\pi}_c \geq 0\}$.

In this paper we will study SVMs based on universal kernels, where $\mathcal{F}$ is a strictly positive definite Reproducing Kernel Hilbert Space (RKHS), $\mathcal{H}_\mathbf{K}$, induced by some known kernel function $K : \mathcal{X} \times \mathcal{X} \mapsto \mathbb{R}$, and endowed by the norm $||.||_{\mathcal{H}_\mathbf{K}}$ (Wahba (1998), Cristianini and Shawe-Taylor (2000), Poggio et al. (2002)). Then, the representer theorem (Kimeldorf and Wahba (1971)) implies that for all $\mathbf{x} \in \mathcal{X}$ and $c \in \mathcal{Y}$, $\mathbf{f_c}(\mathbf{x})$ and $||f_c||^2_{\mathcal{H}_\mathbf{K}}$ can be expressed as $\mathbf{f_c}(\mathbf{x}) = \sum_{i=1}^n \boldsymbol{\theta}_i^c \; K(\mathbf{x_i}, \mathbf{x})$, $||f_c||^2_{\mathcal{H}_\mathbf{K}} = \sum_{i=1}^n \sum_{j=1}^n \boldsymbol{\theta}_i^c \; \boldsymbol{\theta}_j^c \; K(\mathbf{x_i}, \mathbf{x_j})$, $\boldsymbol{\theta}^c \in \mathbb{R}^n$, and the penalty $J(.)$ is typically chosen as the sum of the squared norms of the $\mathbf{f}$ components, *i.e.*, $J(\mathbf{f}) = \sum_{c=1}^k ||\mathbf{f}_c||^2_{\mathcal{H}_\mathbf{K}}$.

We note that this framework differs from the traditional one, often assumed in the SVM literature, in that our classification functions, $\mathbf{f}$, do not include bias terms. Poggio et al. (2002)) show that the general approximation properties of strictly positive definite Reproducing Kernel Hilbert Spaces do not require such terms, and Dogan et al. (2016) recommend dropping them from standard multiclass SVMs, since this suggestion can lead to substantial computational gains without affecting the main statistical properties of the resulting classifiers. We followed Dogan's recommendation and our simulation results, to be described in Section 5, suggest that the resulting weighted SVMs are also not adversely affected by this strategy.

Different weighted versions of known multi-class SVMs can be specified as particular cases of decision rule (2) and optimization model (3) (4), by choosing particular loss functions. For instance, the SVMs proposed by Crammer and Singer (2001) (CS) and Lee, Lin and Wahba (2004) (LLW) are respectively defined by the following losses:

$$L_{CS}(\mathbf{f}, y) = \quad (1 - \max_{c \in \mathcal{Y} \setminus \{y\}} (\mathbf{f}_y - \mathbf{f}_c))_+ \tag{5}$$

$$L_{LLW}(\mathbf{f}, y) \quad = \sum_{c \in \mathcal{Y} \setminus \{y\}} (\mathbf{f}_c + (k-1)^{-1})_+ \tag{6}$$

where $(u)_+ := max(0, u)$.

A desirable theoretical property of weighted multi-class SVMs is weighted Fisher-consistency, which implies that for problems with equal misclassification costs and weighting vectors equal to the a-*priori* class probabilities, as the size of the training sample increases the resulting decision rule approaches the theoretical Bayes rule. Formally, this property may be defined as follows:

Definition 1: Weighted Fisher-consistency (Wu, Zhang and Liu)

A multi-class functional margin based loss is weighted Fisher-consistent if the minimizer $\mathbf{f}^*$ of $E[\boldsymbol{\pi}_Y L(\mathbf{f}(\mathbf{X}), Y) | \mathbf{X} = \mathbf{x})]$ under (4), satisfies

$$argmax_{c \in \mathcal{Y}} \ \mathbf{f}_c^*(\mathbf{x}) = argmax_{c \in \mathcal{Y}} \ \boldsymbol{\pi}_c \ \mathbf{p}_c(\mathbf{x}) \quad \forall \mathbf{x} \in \mathcal{X}, \forall \boldsymbol{\pi} \in A_k$$

Wu et al. (2010) have shown that the loss assumed by Crammer and Singer is not weighted Fisher-consistent, but some truncated versions of it are. In particular, in their simulation studies Wu et al. (2010) used the following Fisher-consistent loss

$$L_{WZL}(\mathbf{f}, y) = max[\max_{c \in \mathcal{Y} \setminus \{y\}} (1 - \mathbf{f}_y - \mathbf{f}_c))_+ \ , \ 1 + (k-1)^{-1}] \tag{7}$$

On the other hand, Lee et al. (2004) proved that for any choice of an $\mathbf{C} \in (\mathbb{R}_0^+)^{k \times k}$ cost matrix , the minimizer of $\sum_{c \in \mathcal{Y} \setminus \{y\}} \mathbf{C}(y, c) \ (\mathbf{f}_c + (k-1)^{-1})_+$ under (4), is given by the vector f^{*1} with components

$$\mathbf{f_j^{*1}} = \begin{cases} 1 & \text{if } j = argmin_{c \in \mathcal{Y}} \sum_{c' \in \mathcal{Y} \setminus \{c\}} \mathbf{C}(c', c) \ p_{c'}(\mathbf{x}) \\ -(k-1)^{-1} & \text{otherwise} \end{cases}$$

In particular, noting that $argmin_{c \in \mathcal{Y}} \sum_{c' \in \mathcal{Y} \setminus \{c\}} \boldsymbol{\pi}_{c'} \ p_{c'}(\mathbf{x}) = argmin_{c \in \mathcal{Y}} \sum_{c' \in \mathcal{Y}} \boldsymbol{\pi}_{c'} \ p_{c'}(\mathbf{x}) - \boldsymbol{\pi}_c \ p_c(\mathbf{x}) = argmax_{c \in \mathcal{Y}} \ \boldsymbol{\pi}_c \ p_c(\mathbf{x})$, the choice of $\mathbf{C}$ as a matrix satisfying $\mathbf{C}(c', c) = \boldsymbol{\pi}_{c'} \ \forall c \neq c'$, leads to the mininizer, f^{*2}, defined as

Table 1: Number of grid points in uniformly distributes grids

$d_{\boldsymbol{\pi}}$	$k=3$	$k=5$	$k=10$	$k=23$
0.05	171	3876	92378	$--$
0.02	1176	211876	2.05×10^9	4.93×10^{13}
0.01	4859	3764376	1.73×10^{12}	5.71×10^{21}

$$\mathbf{f_j^{*2}} = \begin{cases} 1 & \text{if } j = argmax_{c \in \mathcal{Y}} \ \boldsymbol{\pi}_c \ p_c(\mathbf{x}) \\ -(k-1)^{-1} & \text{otherwise} \end{cases}$$

which implies the weighted Fisher-consistency of the Lee, Lin and Wahba loss (6) in the Wu, Zhang and Liu sense.

In this paper we propose to estimate class probabilities from the class predictions given by weighted SVMs based on loss (6). One advantage of using this loss, is that the training of the resulting SVMs leads to convex optimization models, which are faster to solve than the nonconvex models resulting from loss (7). In Section 4 we will describe an efficient algorithm to train the weighted SVMs proposed here.

3 SVM Probability Estimation

In order to estimate class probabilities from the predictions given by weighted SVMs, we need first to train several weighted SVMs with different weight specifications. Let $\mathcal{G}$ be the grid of $\boldsymbol{\pi}$ specifications, and $\boldsymbol{\pi}^g$ a generic element of $\mathcal{G}$. Wu et al. (2010) describe two alternative ways of estimating class probabilities from the predictions given by the $\boldsymbol{\pi}^g$ weighted SVMs. The first method is based on a direct search for the boundaries of A_k where those predictions change. However, this method lead to unstable estimates and was replaced by a second method, where class probabilities are estimated by the $\mathbf{p}^*(\mathbf{x}) = (\mathbf{p}_1^*(\mathbf{x}), \mathbf{p}_2^*(\mathbf{x}), \ldots, \mathbf{p}_k^*(\mathbf{x}))$ solution of the system $h_c(\mathbf{p}_1(\mathbf{x}), \mathbf{p}_2(\mathbf{x}), \ldots, \mathbf{p}_k(\mathbf{x})) = prop_c(\mathbf{x});\ c = 1, 2, \ldots, k$, where $h_c(\mathbf{p}(\mathbf{x}))$ represents the volume proportion of the A_k simplex where $\boldsymbol{\pi}_c \, \mathbf{p}_c(\mathbf{x}) \geq \boldsymbol{\pi}_{c'} \, \mathbf{p}_{c'}(\mathbf{x}) \ \forall c' \neq c$, and $prop_c(\mathbf{x})$ the observed proportion of weighted SVMs that assign an example described by $\mathbf{x}$ to class c. This method leads to consistent, and more stable, estimates of $\mathbf{p}_c(\mathbf{x})$, as long as the $\boldsymbol{\pi}^g$ are uniformly distributed over A_k, and the grid step size, $d_{\boldsymbol{\pi}}$, converges to zero as the sample size increases without limit. However, the requirement of a strict uniform distribution over A_k leads to a grid size of $\#\mathcal{G} = \binom{d_{\boldsymbol{\pi}}^{-1}+k}{k-1}$, a number that increases exponentially with k. Table 1 shows some values of $\#\mathcal{G}$ for different combinations of $d_{\boldsymbol{\pi}}$ and k, up to $k = 23$, which is the number of different classes in character recognition problems. It is clear that this method becomes computationally intractable for problems with moderate or large k, and we are not aware of any application of this approach to problems with more than 5 classes.

In this paper, we will propose an alternative method to recover class probabilities form class predictions, where the grid of weigh specifications does not have to be uniformly

distributed over A_k. Our approach is based on the optimization of an heuristic l_1-norm measure of the agreement between the weighted SVMs predictions and the Bayes classification rules implied by the $\mathbf{p}_c(\mathbf{x})$ estimates. In particular, consider a weight specification $\boldsymbol{\pi}^g$ and the corresponding weighted SVM prediction $\hat{y}_g$. Interpreting $\boldsymbol{\pi}^g$ as a vector of *a-priori* class probabilities, the SVM prediction agrees with the optimal Bayes rule for a vector of *a-posteriori* probabilities, $\mathbf{p}_c(\mathbf{x})$, iff

$$\boldsymbol{\pi}^g_{\hat{y}_g}\mathbf{p}_{\hat{y}_g}(\mathbf{x}) \geq \boldsymbol{\pi}^g_c\,\mathbf{p}_c(\mathbf{x}) \qquad \forall c \in \mathcal{Y}\backslash\{\hat{y}_g\} \tag{8}$$

When it is possible to find $\mathbf{p}(\mathbf{x})$ vectors such that (8) is satisfied for all $\boldsymbol{\pi}^g \in \mathcal{G}$, a reasonable estimation criterion is to choose amongst such vectors, the one that maximizes the desirable sum of the l_1-norm deviations $\boldsymbol{\pi}^g_{\hat{y}_g}\mathbf{p}_{\hat{y}_g}(\mathbf{x}) - \boldsymbol{\pi}^g_c\mathbf{p}_c(\mathbf{x})$. On the other hand, when it is not possible to find a vector that always satisfies (8), one may search for one that minimizes the undesirable deviations $\boldsymbol{\pi}^g_c\mathbf{p}_c(\mathbf{x}) - \boldsymbol{\pi}^g_{\hat{y}_g}\mathbf{p}_{\hat{y}_g}(\mathbf{x})$. Putting these two goals together, we propose to search for the probability estimate that solves

$$\begin{aligned} min_{\mathbf{p}(\mathbf{x})\in A_k} \sum_{\boldsymbol{\pi}^g\in\mathcal{G}} \sum_{c\in\mathcal{Y}\backslash\{\hat{y}_g\}} \quad & \eta\ (\boldsymbol{\pi}^g_c\,\mathbf{p}_c(\mathbf{x}) - \boldsymbol{\pi}^g_{\hat{y}_g}\mathbf{p}_{\hat{y}_g}(\mathbf{x}))_+\ - \\ & -\ (\boldsymbol{\pi}^g_{\hat{y}_g}\mathbf{p}_{\hat{y}_g}(\mathbf{x}) - \boldsymbol{\pi}^g_c\mathbf{p}_c(\mathbf{x}))_+ \qquad (9) \\ subject\ to \qquad & \mathbf{p}_c(\mathbf{x}) \geq\ \epsilon \qquad \forall c \in \mathcal{Y} \qquad (10) \end{aligned}$$

where the constraint (10) enforces strict positivity of the $\mathbf{p}(\mathbf{x})$ estimates, η is an hyper-parameter that controls the trade-off between the desirable and undesirable deviations, and ϵ is a small positive constant. In all the experiments reported here, we set $\epsilon = d_{\boldsymbol{\pi}}/2$, reflecting the numerical precision implied by the grid resolution.

The optimization of l_1-norm measures similar to the one used in (9)-(10) has been widely studied in the Operations Research literature on supervised classification (see Duarte Silva (2017)), where it has been shown that the resulting optimization problems can be easily solved by straightforward linear programming models. In the current context, one important advantage of this method is the fact that it does not require $\boldsymbol{\pi}^g$ to be strictly uniformly distributed over A_k, which allows for alternative ways of defining representative $\mathcal{G}$ sets with a considerable smaller number of grid points. We will describe one such alternative, where we look at one component of $\boldsymbol{\pi}$, say c, at the time, and ensure that a resulting set of $\boldsymbol{\pi}^g$ specifications gives an adequate representation of the $(0,1)$ interval, while the remaining components of $\boldsymbol{\pi}$ are assigned at random. The details are provided in algorithm 1 , where it is shown that the representation of the $(0,1)$ line is satisfied by setting $\boldsymbol{\pi}_c$ in turn to each of the $d_{\boldsymbol{\pi}}^{-1}$-1 uniformly distributed points of the $(0,1)$ line, with a distance of $d_{\boldsymbol{\pi}}$ between each point. The resulting grid size equals $\#\mathcal{G} = k\,d_{\boldsymbol{\pi}}^{-1} - k$, a number that only increases linearly with k. Table 2 shows the number of base SVMs that need trained in this approach. The contrast with the corresponding figures shown before in Table 1 is remarkable.

Summing it up, algorithm 2 summarizes the major steps required to implement the proposal made in this paper.

Algorithm 1 Generating a grid of weight vectors

Parameters: k, $d_{\boldsymbol{\pi}}$

Output: A $\mathcal{G}$ grid of weight vectors

Let $pbc = d_{\boldsymbol{\pi}}^{-1} - 1$.
Let $\mathcal{G} = \emptyset$, $g = 0$.
for $c \in \mathcal{Y}$ **do**
  for $a = 1$ to pbc **do**
    Let $g = g + 1$.
    Initialize the k-dimensional vector $\boldsymbol{\pi}^g$.
    Let $\boldsymbol{\pi}_c^g = a\, d_{\boldsymbol{\pi}}$.
    Generate independently k-1 uniform random numbers ($U_{c'}$, $c' \in \mathcal{Y}\backslash\{c\}$) on the unit interval.
    Let $SU = \sum_{c' \in \mathcal{Y}\backslash\{c\}} U_{c'}$.
    for $c' \in \mathcal{Y}\backslash\{c\}$ **do**
      Let $\boldsymbol{\pi}_{c'}^g = U_{c'}(1 - \boldsymbol{\pi}_c^g)/SU$.
    end for
    Add $\boldsymbol{\pi}^g$ to $\mathcal{G}$.
  end for
end for
return $\mathcal{G}$.

Table 2: Number of grid points in the proposed method

$d_{\boldsymbol{\pi}}$	$k = 3$	$k = 5$	$k = 10$	$k = 23$
0.05	57	95	190	437
0.02	147	245	450	1127
0.01	297	495	990	2277

4 Efficient SVM Training Algorithms

The state of art algorithms for training SVMs are based on decomposition strategies to solve dual formulations of the associated optimization problems. In the case of 2-class problems, a standard reference is Platt's Sequential Minimal Optimization (SMO) algorithm (Platt, 1999) which decomposes a large convex quadratic optimization problem into a sequence of two-dimensional quadratic problems that can be solved analytically. This algorithm was adapted by Dogan, Glasmachers and Igel (2011) to solve the most common k-class SVMs, including those based on the unweighted LLW loss. It is straightforward to show that this approach also applies to SVMs using the weighted LLW loss. In particular, as shown in the Appendix, the Wolfe dual of optimization problem (3) - (4) with loss (6) can be expressed as the following convex quadratic optimization problem with box constrains:

Algorithm 2 Main probability estimation algorithm

Input: A training set $\mathcal{T}$, and a estimation set $\mathcal{E}$

Output: The set $\mathcal{P}$, of probability vectors for $\mathcal{E}$.

```
Let P = ∅.
Define a grid G of weight specifications using algorithm 1.
for π^g ∈ G do
    Solve optimization model (3)-(4) using loss (6).
    for e ∈ E do
        Predict and save a class for entity e based on the π^g weighted SVM.
    end for
end for
for e ∈ E do
    Solve optimization model (9) - (10) and add its solution to P.
end for
return P.
```

$$max_{\boldsymbol{\alpha}\in\mathbb{R}^{n\times k}} \quad \frac{1}{k-1}\sum_{i\in\mathcal{T}}\sum_{c\in\mathcal{Y}\backslash\{y_i\}}\boldsymbol{\alpha}_{ic} - \\ -\frac{1}{2\lambda}\sum_{i,i'\in\mathcal{T}}K(\mathbf{x}_i,\mathbf{x}_{i'})\sum_{c,c'\in\mathcal{Y}}(\delta_{cc'}-\tfrac{1}{k})\,\boldsymbol{\alpha}_{ic}\,\boldsymbol{\alpha}_{i'c'} \tag{11}$$

$$subject\ to \quad 0 \le \boldsymbol{\alpha}_{ic} \le \tfrac{\boldsymbol{\pi}_{y_i}}{n} \qquad i\in\mathcal{T},\ c\in\mathcal{Y}\backslash\{y_i\} \tag{12}$$

$$\boldsymbol{\alpha}_{iy_i} = 0 \qquad i\in\mathcal{T} \tag{13}$$

Problem (11)-(13) can be solved efficiently by the following algorithm: (i) initialize the $\boldsymbol{\alpha}$ vectors at $\mathbf{0}$; (ii) choose the pair of $\boldsymbol{\alpha}$ components, $\boldsymbol{\alpha}_i$ and $\boldsymbol{\alpha}_{i'}$, that lead to the maximal increase in (11) subject to (12)-(13); (iii) find the analytical solution of problem (11)-(13) restricted to $\boldsymbol{\alpha}_i$ and $\boldsymbol{\alpha}_{i'}$; (iv) repeat (ii) and (iii) until convergence.

The details can be found in Dogan et al. (2011), while in Dogan et al. (2016) it is argued that, under reasonable assumptions, the asymptotic time complexity of this algorithm equals the one required to solve Crammer and Singer's SVM. We note that the popularity of Crammer and Singer SVM is mostly due to the fact that this algortithm is genarilly believed to be the fastest to train among all *all-in-one* multiclass SVMs.

5 Simulation Experiments

In this section we illustrate the performance of this proposal, comparing it with five alternative methods under four simulation scenarios previously discussed in the literature. The methods under comparison are our proposal that we will call Probability Vector Machines (PVM), Multinomial Logistic Regression (MLogReg), two pairwise SVM based methods proposed respectively by Wang, Zhang and Wu (WZW) (Wang et al., 2019), and Xu and Wang (XW) (Xu and Wang, 2013), classification trees (TREE) (Breiman et al., 1984), and Random Forests (RF) (Breiman, 2001). The Multinomial Logistic Regression is arguably

the most important reference method in the classical statistical and econometric literature, and relys on mild assumptions about the distribution of the posterior probabilities, $\mathbf{p}_c(\mathbf{x})$. All the remaining methods are distribution free. The pairwise SVM based methods use the well known technique of decomposing a k-class problem into k different
2-class problems based on an *one-against-the-rest* strategy. Classification Trees and Random Forests are two well established machine learning methedologies for supervised classification.

We consider four different experiments with different data conditions. The first two experiments use setups initially considered in Wu et al. (2010). Both these experiments are 3-class problems, and the first one illustrates conditions in which the assumptions of MLogReg are satisfied, while in the second one the data was generated by highly non-linear functions that lead to a gross violation of these assumptions. The third and fourth experiments use setups initially considered in Xu and Wang (2013) where the data was generated by heavy-tailed distribtions that also violate MLogReg assumptions. These two experiments consider, respectivelly a 5-class (3rd experiment) and a 10-class (4th experiment) problem. The details of the data generation are described in the paragraphs below.

Experiment 1. The first experiment uses the data conditions described in Example 1 of Wu et al. (2010), namely training samples of 400 observations with the Y class labels generated uniformly from $\mathcal{Y} = \{1, 2, 3\}$, and 2-dimensional predictors generated conditionally on Y, from a Gaussian distribution with mean vector $\boldsymbol{\mu}(y) = [cos(2y\pi/3), sin(2y\pi/3)]^T$ and covariance matrix $\Sigma = 0.7^2\ \mathbf{I}_2$, with $\mathbf{I}_2$ being the 2-dimensional identity. The left panel of figure 1 displays a validation sample with 1000 examples generated from this condition.

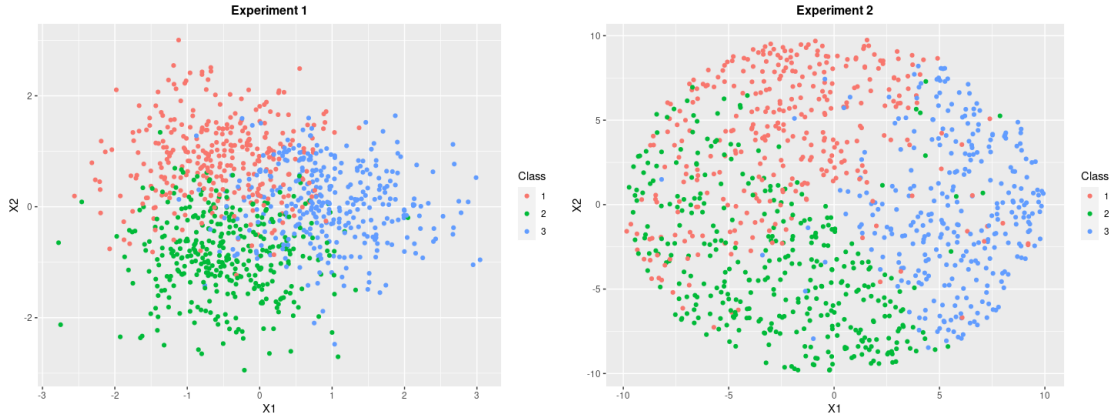

Figure 1: Data for Experiments 1 and 2

Experiment 2. The second experiment uses the data conditions described in Example 3 of Wu et al. (2010), namely training samples of 600 observations with 2-dimensional predictors generated uniformly over the disk $\{\mathbf{x} : x_1^2 + x_2^2 \leq 100\}$, and class probabilities generated conditionally on $\mathbf{x}$ from $p_c(\mathbf{x}) = \exp(g_c(\mathbf{x}))/\sum_{c'=1}^{3}\exp(g_{c'}(\mathbf{x}))\ , c \in \mathcal{Y} = \{1, 2, 3\}$, where $g_1(\mathbf{x}) = \Phi^{-1}(T_2(-5\mathbf{x}_1\sqrt{3} + 5\mathbf{x}_2))$, $g_2(\mathbf{x}) = \Phi^{-1}(T_2(-5\mathbf{x}_1\sqrt{3} - 5\mathbf{x}_2))$, $g_3(\mathbf{x}) = 0$, and

$\Phi(.)$ $T_2(.)$ denote the univariate cumulative standard normal and student t with 2 degrees of freedom (t_2) distributions. The right panel of figure 1 displays a validation sample with 1000 examples generated from this data condition.

Experiments 3 and 4. The third and fourth experiments use the data conditions described in Examples 1 and 2 of Xu and Wang (2013), namely training samples of 400 observations with the Y class labels generated uniformly from $\mathcal{Y} = \{1, 2, \ldots, k\}$, and 2-dimensional predictors generated conditionally on Y, from a t_2 distribution with mean vector $\boldsymbol{\mu}(y) = [cos(2y\pi/k), sin(2y\pi/k)]^T$ and covariance matrix $\Sigma = diag(1, 2)$. In experiment 3, $k = 5$ while in experiment 4, $k = 10$. Figure 2 displays a validation sample with 1000 examples generated from experiment 3. The pattern for the condtion with 10 classes (experiment 4) is similar.

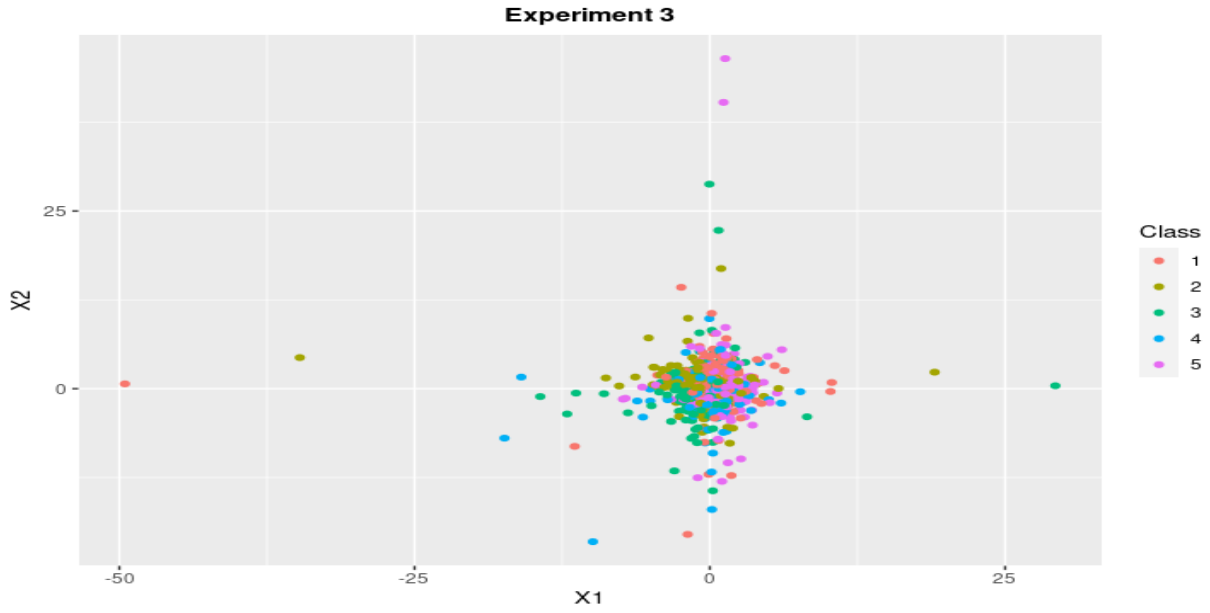

Figure 2: Data for Experiment 3

In all fourth experiments, we trained the six estimation methods on 50 different, independently generated, training samples. In the SVM based methods (PVM, WZW and XW) we always used a Radian Basis Function Kernel, $K(\mathbf{x}_i, \mathbf{x}_{i'}) = exp^{-\rho||\mathbf{x}_i, -\mathbf{x}_{i'}||_2^2}$, with the hyperparameter ρ chosen as the inverse of the median between all pairwise distances $||\mathbf{x}_i, -\mathbf{x}_{i'}||_2^2 \quad i, i' \in \mathcal{T} \quad i \neq i'$, in the training sample (see Caputo et al. (2002)). The regularization parameter λ in (3), was found by a two step search that tries to minimize the log-likelihood, $\ln L = \sum_i \ln \hat{\mathbf{p}}_{y_i}(\mathbf{x}_i)$, in an independently generated tuning set with the same size as the training set. In the first step we searched for the λ_0 value that minimizes $\ln L$ over the set $\{2^{5j} | \, j = -3, -2, \ldots, 3\}$, and in the second step we refined the search, looking for the the minimizer of $\ln L$ over $\{2^j \lambda_0 | \, j = -2, -1, 0, 1, 2\}$. For the PVM method, the grid step size was set at $d_{\boldsymbol{\pi}} = 0.2/\sqrt{n}$ which leads to $d_{\boldsymbol{\pi}} = 0.01$ in experiments 1, 3 and 4,

and $d_{\boldsymbol{\pi}} = 0.0082$ in experiment 2. The η hyperparameter in (9) was alwyas set at $\eta = 15$ which, in auxiliary simulation experiments (results not shown), has proved to lead to good results across a wide range of different data conditions. For the Tree and Random Forest methods we relied on the *TREE* and *RF* functions of the *rpart* (Therneau and Atkinson, 2019) and *randomForest* (Liaw and Wiener, 2002) R packages, with all arguments set at their default values,

For all experiments and replications, we evaluated the six estimation methods under comparison, by computing the l_2 norm error, $l_2\, err = \frac{1}{\#\mathcal{E}} \sum_{i \in \mathcal{E}} \sum_{c \in \mathcal{Y}} (\hat{\mathbf{p}}_c(\mathbf{x}_i) - \mathbf{p}_c(\mathbf{x}_i))^2$, and Empirical Generalized Kullback-Leiber (EGKL) loss, $EGKL = \frac{1}{\#\mathcal{E}} \sum_{i \in \mathcal{E}} \sum_{c \in \mathcal{Y}} \mathbf{p}_c(\mathbf{x}_i) \ln \frac{\mathbf{p}_c(\mathbf{x}_i)}{\hat{\mathbf{p}}_c(\mathbf{x}_i)}$, in an independently generated validation data set ($\mathcal{E}$) with 1000 examples.

Figures 3 through 6 show sinaplots (Sidiropoulos et al., 2018) of the l_2-norm and EGKL errors for all simulation experiments, while tables 3 through 6 present the corresponding means, mean standard errors, and medians. Tables 3 and 4 also show, under the WZL acrynom, the reported average of these measures (see Wu et al. (2010)) for the original Wu, Zhang and Liu proposal. We note that, unlike in our experiments, Wu et al. (2010) did not use an universal kernel but a linear one, taking advantage of the knownn form of the classification boundaries for these conditions. However, in real data problems the true form of these boundaries is always unknown.

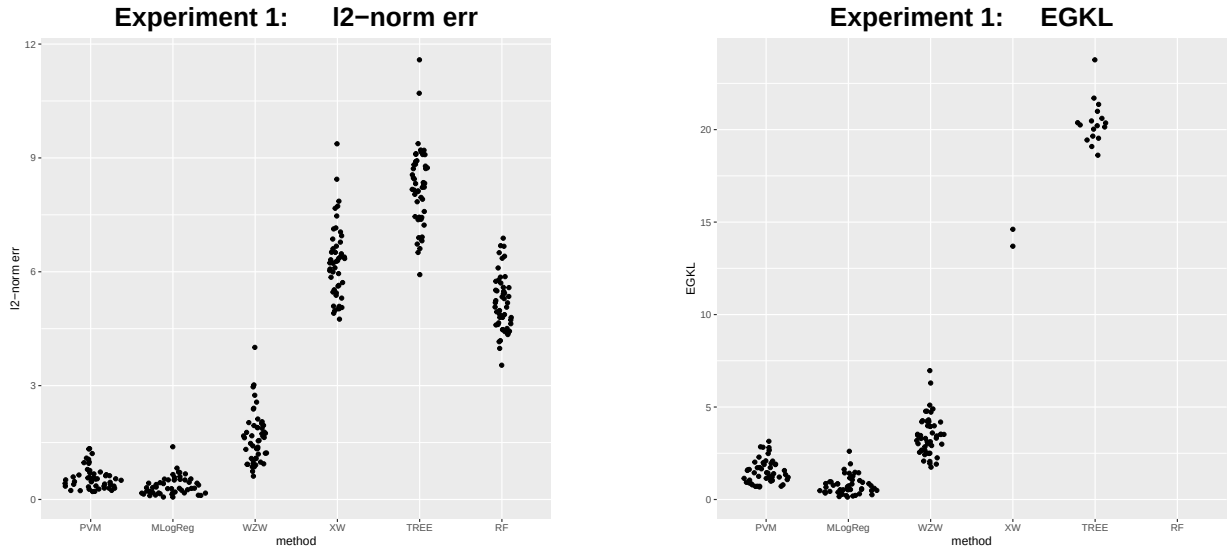

Figure 3: l2-norm and EGKL error rates of Experiment 1.

For some combinations of experiments, replications, and classes, the XL, TREE or RF methods estimate class probabilities exactly by 0 which leads to an infinite value of the EGKL loss. This problem does not occur for the l_2-norm error measure, nor for the

Table 3: Error rates for Example 1

	PVM	MLogReg	WZW	XW	TREE	RF	WZL
l2err							
mean	0.57	0.36	1.62	6.26	8.23	5.18	0.90
(stderr)	(0.04)	(0.03)	(0.09)	(0.13)	(0.15)	(0.11)	–
median	0.52	0.30	1.59	6.26	8.26	5.13	–
EGKL							
mean	1.53	0.77	3.47	∞	∞	∞	2.56
(stderr)	(0.09)	(0.07)	(0.15)	–	–	–	–
median	1.44	0.64	3.38	∞	∞	∞	–

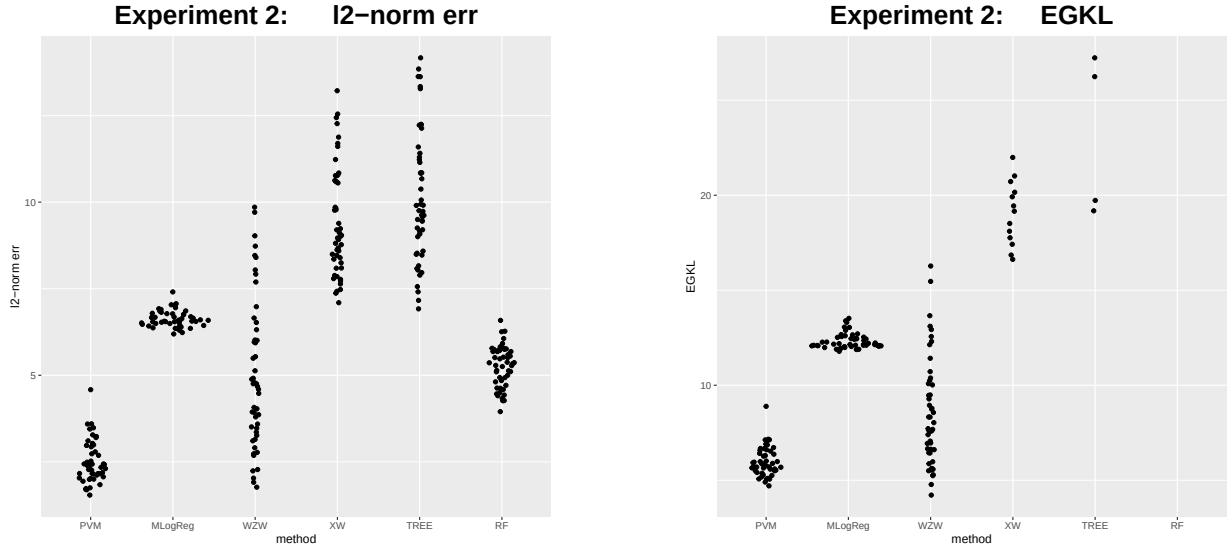

Figure 4: l2-norm and EGKL error rates of Experiment 2.

Table 4: Error rates for Example 2

	PVM	MLogReg	WZW	XW	TREE	RF	WZL
l2err							
mean	2.50	6.62	5.01	9.44	10.13	5.26	4.47
(stderr)	(0.09)	(0.03)	(0.31)	(0.22)	(0.27)	(0.08)	–
median	2.38	6.58	4.71	9.03	9.75	5.35	–
EGKL							
mean	5.97	12.33	8.49	∞	∞	∞	11.79
(stderr)	(0.11)	(0.06)	(0.40)	–	–	–	–
median	5.81	12.18	7.70	∞	∞	∞	–

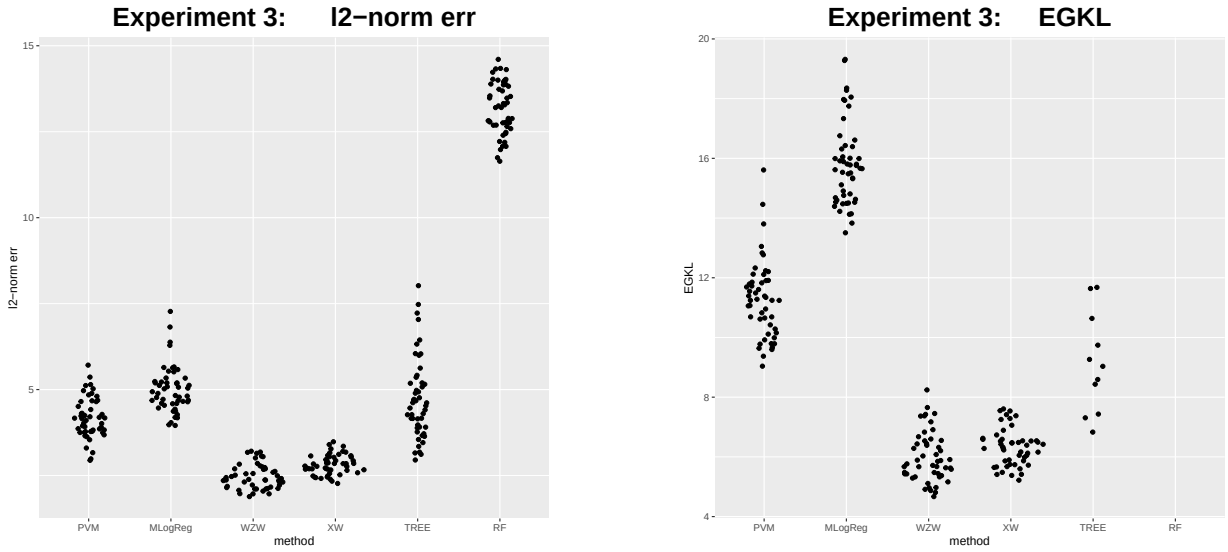

Figure 5: l2-norm and EGKL error rates of Experiment 3.

Table 5: Error rates for Experiment 3

	PVM	MLogReg	WZW	XW	TREE	RF
l_2 err						
mean	4.17	5.01	2.47	2.82	4.73	13.19
(stderr)	(0.08)	(0.10)	(0.05)	(0.04)	(0.17)	(0.11)
median	4.13	4.91	2.39	2.81	4.57	13.23
EGKL						
mean	11.28	15.80	6.00	∞	∞	∞
(stderr)	(0.18)	(0.20)	(0.12)	–	–	–
median	11.26	15.66	5.81	6.36	∞	∞

Table 6: Error rates for Experiment 4

	PVM	MLogReg	WZW	XW	TREE	RF
l_2 err						
mean	2.63	3.09	1.81	1.81	3.93	14.89
(stderr)	(0.06)	(0.05)	(0.03)	(0.03)	(0.13)	(0.10)
median	2.53	3.07	1.81	1.78	3.82	14.81
EGKL						
mean	11.83	18.47	8.64	8.03	∞	∞
(stderr)	(0.19)	(0.29)	(0.16)	(0.15)	–	–
median	11.42	18.14	8.54	7.76	∞	∞

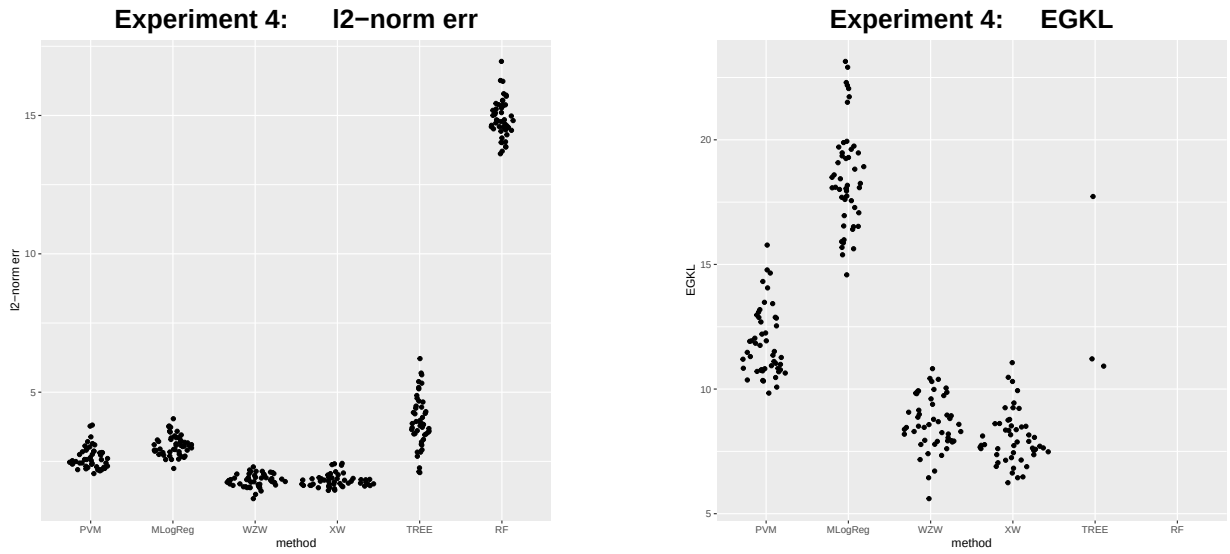

Figure 6: l2-norm and EGKL error rates of Experiment 4.

remaining three methods, which always enforce strictly positive probability estimates. Nevertheless, we prefer the EGKL loss as an global measure of estimation performance because this measure was specifically designed to compare probability distributions and, unlike the l_2 norm error, does not treat equal absolute errors as equally important, regardless of being associated with probabilities close to 0.5 or to the 0. or 1. boundaries of their domain. However, we note that, in these experiments, when results for the EGKL loss are available the resulting rankings of the six estimation methods tends to agree with the rankings given by the l_2 error.

Overall, these results confirm the previous findings in this literature. In particular, multinomial logistic regression gives the best results when its assumptions are met (Experiment 1), but when they are grossly violated one of the SVM based methods performs the best. On the other hand, the performance of the Tree and Random Forest methods is disappointing, suggesting that in spite of their merits for pure supervised classification problems, they are not as reliable at providing estimates of class probabilities. Among the three SVM based methods we did not find a single clear overall winner. In these experiments, the pairwise *one-against-all* methods performed the best for the conditions characterized by heavy-tailed distributions (Experiments 3 and 4), while the *all-in-one* approach studied here gave the best results for the non-lineary transformed data (Experiment 2). Furthermore, this approach seems relatively stable across different data conditions, coming often as second best when it is not the ideal method and, in particular, performing better than the SVM pairwise methods whem the assumptions of logistic regression are satisfied (Experiment 1), and better than logistic regression when the data has severe outliers (Experiments 3 and 4). Among the parwise *one-against-all* SVM methods the proposal of Wang, Zhang and Wu performed better the method of Xu and Wang, confirming previous findings reported in Wang et al. (2019). Finally the comparison of the results of experiments 3 and 4 suggests that the number of different classes might not a strong influence on the relative

standing of alternative estimation methods. Naturally, additional studies are necessary to verify if this result holds for other, more general, data conditions.

Remarkably, in the experiments for which results for the original *all-in-one* Wu et al. (2010) proposal were available, those results were improved by the method developed here, even though we used an universal agnostic kernel, and only 297 (Experiment 1) and 364 (Experiment 2) grid points, while Wu et al. (2010) used the ideal kernel for these conditions, and a total of 1176 grid points in each experiment. We suspect that this surprising result might be explained by the fact the recovery of probabilities by the optimizing (9) uses more information than the matching of observed with expected frequencies employed in the original Wu et al. (2010) proposal.

6 Conclusions

Kernel based methods are an important set of tools for any modern statistician or econometrician. However, most common kernel methods focus on pure prediction problems, and do not pay enough attention to the related, and critical, problem of providing reliable confidence measures for those predictions. In the particular case of kernel based classification SVMs, this problem has being tackled by taking advantage of information provided by sequences of weighted SVMs. However, up to now, for a moderate or large number of different classes this approach was only computationally feasible using pairwise *one-against-all* strategies, and not by the theoretically more sound *all-in-one* approach.

In this paper, we have filled this gap, and developed a computationally efficient estimation method based on sequences of weighted SVMs that consider all classes, simultaneously. Furthermore, we have provided statistical evidence that SVM based probability estimators are among the most reliable distribution free estimators for class probabilities, and beat multinomial logistic regression when its assumptions are grossly violated. In line with similar results for the pure classification problems, we have found that no particular method of extending 2-class to general k-class SVM methodologies dominates the alternatives, and that different data conditions may favor different approaches.

A public domain R package implementing the methods discussed here is under preparation, and we expect to submit it to the CRAN repository in the not too distant future.

References

Breiman, L. (2001). Random forests. *Machine Learning 45*(1), 5–32.

Breiman, L., J. Friedman, R. Olshen, and C. Stone (1984). *Classification and Regression Trees*. Wadsworth Publishing Company.

Caputo, B., K. Sim, F. Furesjo, and A. Smola (2002). Appearance-based object recognition using svms: which kernel should i use? In *Proceedings of the Neural Information Processing Systems Workshop on Statistical Methods for Computational Experiments in Visual Processing and Computer Vision*. Whistler.

Crammer, C. and Y. Singer (2001). On the algorithmic implementation of multiclass kernel-based vector machines. *Journal of Machine Learning Research 2*, 265–292.

Cristianini, N. and J. Shawe-Taylor (2000). *An introduction to Support Vector Machines*. Cambridge University Press.

Dogan, U., T. Glasmachers, and C. Igel (2011). Fast training of multi-class support vector machines. Technical report, Department of Computer Science, University of Copenhagen.

Dogan, U., T. Glasmachers, and C. Igel (2016). A unified view on multi-class support vector classification. *Journal of Machine Learning Research 17*(45), 1–32.

Duarte Silva, A. P. (2017). Optimization approaches to supervised classification. *European Journal of Operational Research 261*(2), 772–788.

Kimeldorf, G. and G. Wahba (1971). Somes results on tchebycheffian spline functions. *Journal of Mathematical Analysis and Applications 33*, 82–95.

Lee, Y., Y. Lin, and G. Wahba (2004). Multicategory support vector machines: Theory and application to the classification of microarray data and satellite radiance data. *Journal of the American Statistical Association 99*(465), 67–81.

Liaw, A. and M. Wiener (2002). Classification and regression by randomforest. *R News 2*(3), 18–22.

Lin, Y. (2002). Support vector machines and the bayes rule in classification. *Data Mining and Knowledge Discovery 6*(3), 988–994.

Platt, J. (1999). Fast training of support vector machines using sequential minimal optimization. In B. Schölkopf, J. Burges, C, and A. Somola (Eds.), *Advances in kernel methods—support vector learning*, pp. 185–208. MIT Press.

Poggio, T., S. Mukherjee, R. Rifkin, A. Raklin, and A. Verri (2002). B. In J. Winkler and M. Niranjan (Eds.), *Uncertainty in Geometric Computations*, pp. 131–141. Springer US.

Sidiropoulos, N., S. Sohi, T. Pedersen, B. Porse, O. Winther, N. Rapin, and F. Bagger (2018). Sinaplot: an enhanced chart for simple and truthful representation of single observations over multiple classes. *Journal of Computational and Graphical Statistics 27*(3), 673–676.

Therneau, T. and B. Atkinson (2019). *rpart: Recursive Partitioning and Regression Trees*. R package version 4.1-15.

Wahba, G. (1998). Support vector machines, reproducing kernel hilbert spaces, and randomized gacv. In B. Schölkopf, C. Burges, and A. Somola (Eds.), *Advances in Kernel Methods: Support Vector Learning*, pp. 69–87. MIT Press.

Wang, J., X. Shen, and L. Yufeng (2008). Probability estimation for large-margin classifiers. *Biometrika 95*(1), 149–167.

Wang, X., H. Zhang, and W. Yichao (2019). Multiclass probability estimation with support vector machines. *Journal of Computational and Graphical Statistics 28*(3), 586–595.

Wu, Y., H. Zhang, and Y. Liu (2010). Probability estimation for large-margin classifiers. *Journal of the American Statistical Association 105*(489), 424–436.

Xu, T. and J. Wang (2013). An efficient model-free estimation of multiclass conditional probability. *Journal of Statistical Planning and Inference 143*, 2079–2088.

Appendix

Derivation of SVM dual

Here, we will show that the dual of optimization problem (3) - (4) with loss (6) is given by problem (11) - (13). First notice that, using matrix notation and introducing slack variables, the primal problem can be expressed as

$$
\begin{aligned}
min_{\boldsymbol{\theta},\boldsymbol{\epsilon}\in\mathbb{R}^{n\times k}} \quad & \textstyle\sum_{c\in\mathcal{Y}} \left[\lambda\ \boldsymbol{\theta}_{.c}^T\ \mathbf{K}\ \boldsymbol{\theta}_{.c}\ +\ \mathbf{C}_{.c}^T\ \boldsymbol{\epsilon}_{.c}\right] && (14)\\
\textit{subject to} \quad & \mathbf{K}\ \boldsymbol{\theta}\ \leq\ -(k-1)^{-1}\ \mathbf{1}_{n\times k} + \boldsymbol{\epsilon} && (15)\\
& \textstyle\sum_{c\in\mathcal{Y}}\ \mathbf{K}\ \boldsymbol{\theta}_{.c} =\ \mathbf{0}_n && (16)\\
& \boldsymbol{\epsilon}\ \geq\ \mathbf{0}_{n\times k} && (17)
\end{aligned}
$$

where, with a slight abuse of notation, the matrix $\boldsymbol{\theta}$ is defined as the matrix with columns $\boldsymbol{\theta}_{.c}$ equal to the $\boldsymbol{\theta}^c$ vectors of the $\mathbf{f}_c$ expansion given by the representer theorem, $\mathbf{K} \in \mathbb{R}^{n\times n}$ is the Gramm matrix with generic element $\mathbf{K}_{ii'} = K(\mathbf{x}_i, \mathbf{x}_{i'})$, and the cost matrix $\mathbf{C} \in \mathbb{R}^{n\times k}$ has generic elements $\mathbf{C}_{ic} = n^{-1}\ \pi_{y_i}$ if $c \neq y_i$, and $\mathbf{C}_{iy_i} = 0$. Furthermore, we denote the transpose of a generic matrix $\mathbf{M}$ (or vector $\mathbf{v}$) by $\mathbf{M}^T$ ($\mathbf{v}^T$), its c-column by $\mathbf{M}_{.c}$ and $\mathbf{1}_{n\times k},\ \mathbf{0}_{n\times k},\ \mathbf{0}_n$ are $n \times k$ matrices and an n-dimensional vector with all elements equal, respectively, to 1 and 0.

Introducing Lagrange multipliers for constraints (15), (16) and (17), the Wolfe dual problem can be written as

$$
\begin{aligned}
& max_{\boldsymbol{\alpha},\boldsymbol{\gamma}\in\mathbb{R}^{n\times k},\boldsymbol{\beta}\in\mathbb{R}^n}\ \inf_{\boldsymbol{\theta},\boldsymbol{\epsilon}\in\mathbb{R}^{n\times k}} L_D = \\
& = \textstyle\sum_{c\in\mathcal{Y}} \left[\lambda\ \boldsymbol{\theta}_{.c}^T\ \mathbf{K}\ \boldsymbol{\theta}_{.c} + \mathbf{C}_{.c}^T\ \boldsymbol{\epsilon}_{.c} +\ \boldsymbol{\alpha}_{.c}^T\ \left(\mathbf{K}\ \boldsymbol{\theta}_{.c} + \tfrac{1}{k-1}\ \mathbf{1}_n - \boldsymbol{\epsilon}_{.c}\right) - \boldsymbol{\gamma}_{.c}^T\ \boldsymbol{\epsilon}_{.c}\right]\ + \\
& \qquad + \boldsymbol{\beta}^T\ \left[\textstyle\sum_{c\in\mathcal{Y}}\ \mathbf{K}\ \boldsymbol{\theta}_{.c}\right] && (18)\\
& \textit{subject to} \qquad \boldsymbol{\alpha},\ \boldsymbol{\gamma}\ \geq\ \mathbf{0}_{n\times k} && (19)
\end{aligned}
$$

Minimization in order to the $\boldsymbol{\theta}$ and $\boldsymbol{\epsilon}$ primal variables leads to

$$
\begin{aligned}
& \frac{\partial L_D}{\partial \boldsymbol{\theta}_{.c}}\ = \mathbf{0_n}\ \Leftrightarrow\ 2\lambda\ \boldsymbol{\theta}_{.c}^T\ \mathbf{K}\ + (\boldsymbol{\alpha}_{.c}^T + \boldsymbol{\beta}^T)\ \mathbf{K}\ = \mathbf{0_n}\ \Rightarrow\ \boldsymbol{\theta}_{.c}\ =\ -(2\lambda)^{-1}\ (\boldsymbol{\alpha}_{.c} + \boldsymbol{\beta}) && (20)\\
& \frac{\partial L_D}{\partial \boldsymbol{\epsilon}_{.c}}\ = \mathbf{0_n}\ \Leftrightarrow\ \mathbf{C}_{.c}\ - (\boldsymbol{\alpha}_{.c} + \boldsymbol{\gamma}) = \mathbf{0}_n && (21)
\end{aligned}
$$

where the second equality in (20) follows from the fact that $\mathbf{K}$, being a positive definite matrix, is always non-singular.

From (21) and (19), it follows that

$$\mathbf{0_{n\times k}} \;\leq\; \boldsymbol{\alpha} \;\leq\; \mathbf{C} \tag{22}$$

while replacing (20) into (16) leads to

$$\sum_{c\in\mathcal{Y}} \mathbf{K}\;(\boldsymbol{\alpha}_{.c} + \boldsymbol{\beta}) = \mathbf{0_n} \;\Rightarrow\; \boldsymbol{\beta} \;=\; -k^{-1} \sum_{c\in\mathcal{Y}} \boldsymbol{\alpha}_{.c} \tag{23}$$

Replacing (20) and (23) into (18), and simplifying with the help of (21), leads to

$$L_D = \sum_{c\in\mathcal{Y}} [\;(k-1)^{-1}\;\boldsymbol{\alpha}_{.c}^T\;\mathbf{1_n} \;-\; (2\lambda)^{-1}\;\;\boldsymbol{\alpha}_{.c}^T\;\mathbf{K}\;(\boldsymbol{\alpha}_{.c} \;-\; k^{-1}\;\sum_{c'\in\mathcal{Y}} \boldsymbol{\alpha}_{.c'})\;] \tag{24}$$

so that the dual in matrix form is given by

$$\begin{array}{l} max_{\boldsymbol{\alpha}\in\mathbb{R}^{n\times k}}\; \sum_{c\in\mathcal{Y}} [\;(k-1)^{-1}\;\boldsymbol{\alpha}_{.c}^T\;\mathbf{1_n} \;-\; (2\lambda)^{-1}\;\;\boldsymbol{\alpha}_{.c}^T\;\mathbf{K}\;(\boldsymbol{\alpha}_{.c} \;-\; k^{-1}\;\sum_{c'\in\mathcal{Y}} \boldsymbol{\alpha}_{.c'})\;] \\ \quad \textit{subject to} \qquad \mathbf{0_{n\times k}} \;\leq\; \boldsymbol{\alpha} \;\leq\; \mathbf{C} \end{array}$$

which is equivalent to (11) - (13).